CLAIMS

What is claimed is:

1. A method for maintaining data integrity, comprising the computer-implemented steps of:
generating checksum data by performing a physical checksum calculation on a block of data in volatile memory;
after generating said checksum data,
performing a logical check on data contained within the block of data; and
if the block of data passes said logical check, then causing the block of data to be written to nonvolatile memory.

2. The method of Claim 1 wherein the steps of generating checksum data and performing a logical check are performed in response to a request to write said block of data to nonvolatile memory.

3. The method of Claim 1 further comprising the step of writing the checksum data to nonvolatile memory in association with writing said block of data to nonvolatile memory.

4. The method as recited in Claim 3, further comprising the steps of:
after writing the block of data to nonvolatile memory,
causing the block of data and said checksum data to be read from nonvolatile memory; and
performing a physical checksum verification procedure on said block of data based on said checksum data, wherein the physical checksum verification procedure indicates whether the block of data was corrupted subsequent to performing the logical check on the data contained with the block of data.

5. The method as recited in Claim 1, further comprising the step of performing one or more physical checksum verification procedures prior to writing the block of data to nonvolatile memory, wherein the one or more physical checksum verification procedures indicate whether the block of data was corrupted subsequent to generating said checksum data.

6. The method as recited in Claim 1, wherein:
the step of performing a physical checksum calculation comprises the step of a software application performing the physical checksum calculation on said block of data; and
the step of performing a logical check on data contained with the block of data comprises the step of said software application performing the logical check on data contained with the block of data.

7. The method as recited in Claim 4, wherein:
the step of performing a physical checksum calculation comprises the step of an software application performing the physical checksum calculation on said block of data;
the step of performing a logical check on data contained with the block of data comprises the step of said software application performing the logical check on data contained with the block of data; and
the step of performing a physical checksum verification procedure on said block of data comprises the step of said software application performing the physical checksum verification procedure on said block of data.

8. The method as recited in Claim 5, wherein:
the step of performing a physical checksum calculation comprises the step of an software application performing the physical checksum calculation on said block of data;

the step of performing a logical check on data contained with the block of data comprises the step of said software application performing the logical check on data contained with the block of data; and

the step of performing one or more physical checksum verification procedures prior to writing the block of data to nonvolatile memory comprises the step of one or more components other than said software application performing the one or more physical checksum verification procedures prior to writing the block of data to nonvolatile memory.

9. The method as recited in Claim 4, further comprising the step of:

after performing the physical checksum verification procedure on said block of data, storing the block of data as a backup version of the block of data, wherein the backup version of the block of data is maintained separate from said block of data in said nonvolatile memory.

10. A method for storing data in a nonvolatile memory, comprising the computer-implemented steps of:

determining a desired location in said nonvolatile memory for storing a data block;

inserting an address value in said data block, wherein the address value identifies the desired location;

prior to performing an operation that stores the data block to nonvolatile memory, verifying that the address value contained within the data block correctly identifies the location in nonvolatile memory into which the operation is going to store the data block; and

performing the operation to store the data block to nonvolatile memory only if the address value contained within the data block correctly identifies the location in nonvolatile memory into which the operation is going to store the data block.

11. The method as recited in Claim 10, further comprising the steps of:
after storing the block of data to nonvolatile memory,
reading the block of data from a location in nonvolatile memory; and
comparing the address value contained within the data block with the location in nonvolatile memory from which the data block was read.

12. The method as recited in Claim 10, further comprising the step of maintaining a mapping that identifies a specific location in said nonvolatile memory into which said data block is to be stored.

13. The method as recited in Claim 10, wherein:
the step of determining a location in said nonvolatile memory comprises the step of determining a plurality of locations in said nonvolatile memory for storing said data block;
the step of inserting an address value in said data block, comprises the step of inserting a plurality of address values in said data block, wherein the plurality of address values identify multiple locations in said nonvolatile memory for which the data block is to be stored; and
the step of storing the data block to nonvolatile memory comprises the step of storing the data block in each of the multiple locations in nonvolatile memory only after verifying the plurality of address values includes an address value that correctly identifies the location in nonvolatile memory into which the data block is to be stored.

14. A method for maintaining data integrity, comprising the computer-implemented steps of:
performing a physical checksum calculation on a block of data;
after performing the physical checksum calculation,
performing a first physical checksum verification procedure on said block of data prior to writing the block of data to nonvolatile memory,

wherein the first physical checksum verification procedure indicates whether the block of data was corrupted subsequent to performing the physical checksum calculation on the data contained with the block of data; and

if the block of data passes said first physical checksum verification procedure, then causing the block of data to be written to nonvolatile memory.

15. The method as recited in Claim 14, further comprising the steps of:

after writing the block of data to nonvolatile memory,

causing the block of data to be read from nonvolatile memory; and

performing a second physical checksum verification procedure on said block of data, wherein the second physical checksum verification procedure indicates whether the block of data was corrupted subsequent to performing the first physical checksum verification procedure on the data contained with the block of data.

16. The method as recited in Claim 14, wherein the step of performing a first physical checksum verification procedure includes the steps of performing a plurality of physical checksum verification procedures on said block of data prior to writing the block of data to nonvolatile memory, wherein the plurality of physical checksum verification procedures indicate whether the block of data was corrupted subsequent to performing the physical checksum calculation on the data contained with the block of data.

17. The method as recited in Claim 14, wherein:

the step of performing a physical checksum calculation comprises the step of a software application performing the physical checksum calculation on said block of data; and

the step of performing a first physical checksum verification procedure on said block of data comprises the step of a component other than said software

application performing said first physical checksum verification procedure on said block of data prior to writing the block of data to nonvolatile memory.

18. The method as recited in Claim 17, wherein the step of performing a physical checksum verification procedure on said block of data comprises the step of a disk array component performing the physical checksum verification procedure on said block of data, wherein the disk array component is configured to write the block of data to disk only after verifying the integrity of the data block.

19. The method as recited in Claim 14, further comprising the step of:
after performing the physical checksum calculation,
performing a logical check on data contained with the block of data; and
if the block of data does not pass said logical check, then not writing the block of data to nonvolatile memory.

20. The method as recited in Claim 19, further comprising the step:
the step of performing a physical checksum calculation comprises the step of an software application performing the physical checksum calculation on said block of data;
the step of performing a logical check on data contained with the block of data comprises the step of said software application performing the logical check on data contained with the block of data; and
the step of performing a first physical checksum verification procedure prior to writing the block of data to nonvolatile memory comprises the step of one or more components other than said software application performing one or more physical checksum verification procedures prior to writing the block of data to nonvolatile memory.